Application Serial No. 10/037,390
Group Art Unit 2124

In the claims:

Please amend the claims as follows:

106.   (Currently Amended) A microcontroller comprising:

a memory storing:

a derivative application derived from an application having a class file format wherein the application is derived from an application having a class file format by first compiling the application having a class file format into a compiled form and then converting the compiled form into a converted form, and

an interpreter configured to interpret derivative applications in the converted form and derived from applications having a class file format; and

a processor coupled to the memory, the processor configured to use the interpreter to interpret the derivative application for execution.

107.   (Previously Added) The microcontroller of claim 106, further comprising:
a communicator configured to communicate with a terminal.

108.   (Currently Amended) The microcontroller of claim 107, wherein the terminal has a card reader and the communicator comprises a contact for communicating with the card reader.

109.   (Previously Added) The microcontroller of claim 108, wherein the terminal has a wireless communicator and a wireless transceiver for communicating with the wireless communication device.

110.   (Previously Added) The microcontroller of claim 108, wherein the terminal has a wireless communication device and the communicator comprises a wireless transmitter for communicating with the wireless communication device.

111.   (Previously Added) The microcontroller of claim 106, wherein the class file format comprises a Java class file format.

2

Application Serial No. 10/037,390
Group Art Unit 2124

112.   (Previously Added) A microcontroller having a set of resource constraints and comprising:

    a memory, and

an interpreter loaded in memory and operable within the set of resource constraints, the microcontroller having: at least one application loaded in the memory to be interpreted by the interpreter, wherein the at least one application is generated by a programming environment comprising:

    a) a compiler for compiling application source programs written in high level language source code form into a compiled form, and

    b) a converter for post processing the compiled form into a minimized form suitable for interpretation within the set of resource constraints by the interpreter.

113.   (Previously Added) The microcontroller of Claim 112, wherein the compiled form includes attributes, and the converter comprises a means for including attributes required by the interpreter while not including the attributes not required by the interpreter.

114.   (Previously Added) The microcontroller of Claim 112 wherein the compiled form is in a standard Java class file format and the converter accepts as input the compiled form in the standard Java class file format and produces output in a form suitable for interpretation by the interpreter.

115.   (Previously Added) The microcontroller of Claim 112 wherein the compiled form includes associating an identifying string for objects, classes, fields, or methods, and the converter comprises a means for mapping such strings to unique identifiers.

116.   (Previously Added) The microcontroller of Claim 115 wherein each unique identifier is an integer.

3

Application Serial No. 10/037,390
Group Art Unit 2124

117.   (Previously Added) The microcontroller of Claim 115 wherein the mapping of strings to unique identifiers is stored in a string to identifier map file.

118.   (Previously Added) The microcontroller of Claim 112 where in the high level language supports a first set of features and a first set of data types and the interpreter supports a subset of the first set of features and a subset of the first set of data types, and wherein the converter verifies that the compiled form only contains features in the subset of the first set of features and only contains data types in the subset of the first set of data types.

119.   (Previously Added) The microcontroller of Claim 115 wherein the compiled form is in a byte code format and the converter comprises means for translating from the byte codes in the compiled form to byte codes in a format suitable for interpretation by the interpreter by:

using at least one step in a process including the steps:

a) recording all jumps and their destinations in the original byte codes;

b) converting specific byte codes into equivalent generic byte codes or vice-versa;

c) modifying byte code operands from references using identifying strings to references using unique identifiers; and

d) renumbering byte codes in the compiled form to equivalent byte codes in the format suitable for interpretation; and

relinking jumps for which destination address is effected by conversion step a), b), c), or d).

4

Application Serial No. 10/037,390
Group Art Unit 2124

120.   (Previously Added) The microcontroller of Claim 112 wherein the application program is compiled into a compiled form for which resources required to execute or interpret the compiled form exceed those available on the microcontroller.

121.   (Previously Added) The microcontroller of Claim 112 wherein the compiled form is designed for portability on different computer platforms.

122.   (Previously Added) The microcontroller of Claim 112 wherein the interpreter is further configured to determine, during an interpretation of an application, whether the application meets a security criteria selected from a set of rules containing at least one rule selected from the set:

    not allowing the application access to unauthorized portions of memory,

    not allowing the application access to unauthorized microcontroller resources,

    wherein the application is composed of byte codes and checking a plurality of byte codes at least once prior to execution to verify that execution of the byte codes does not violate a security constraint.

123.   (Previously Added) The microcontroller of Claim 112 wherein at least one application program is generated by a process including the steps of:

    prior to loading the application verifying that the application does not violate any security constraints; and

    loading the application in a secure manner.

124.   (Previously Added) The microcontroller of Claim 123 wherein the step of loading in a secure manner comprises the step of:

verifying that the loading identity has permission to load applications onto the microcontroller.

125.   (Previously Added) The microcontroller of Claim 123 wherein the step of loading in a secure manner comprises the step of:

encrypting the application to be loaded using a loading key.

5

Application Serial No. 10/037,390
Group Art Unit 2124

126. (Previously Added) A method of programming a microcontroller having a memory and a processor operating according to a set of resource constraints, the method comprising the steps of:

inputting an application program in a first programming language;

compiling the application program in the first programming language into a first intermediate code associated with the first programming language, wherein the first intermediate code being interpretable by at least one first intermediate code virtual machine;

converting the first intermediate code into a second intermediate code; wherein the second intermediate code is interpretable within the set of resource constraints by at least one second intermediate code virtual machine; and

loading the second intermediate code into the memory of the microcontroller.

127. (Previously Added) The method of programming a microcontroller of Claim 126 wherein the step of converting further comprises:
associating an identifying string for objects, classes, fields, or methods; and mapping such strings to unique identifiers.

128. (Previously Added) The method of Claim 127 wherein the step of mapping comprises the step of mapping strings to integers.

129. (Previously Added) The method of Claim 126 wherein the step of converting comprises at least one of the steps of:

a) recording all jumps and their destinations in the original byte codes;

b) converting specific byte codes into equivalent generic byte codes or vice-versa;

c) modifying byte code operands from references using identifying strings to references using unique identifiers;

d) renumbering byte codes in a compiled format to equivalent byte codes in a format suitable for interpretation; and

6

e) relinking jumps for which destination address is effected by conversion step a), b), c), or d).

130.   (Previously Added) The method of Claim 126 wherein the step of loading the second intermediate code into the memory of the microcontroller further comprises checking the second intermediate code prior to loading the second intermediate code to verify that the second intermediate code meets a predefined integrity check and that loading is performed according to a security protocol.

131.   (Previously Added) The method of Claim 130 wherein the security protocol requires that a particular identity must be validated to permit loading prior to the loading of the second intermediate code.

132.   (Previously Added) The method of Claim 130 further characterized by providing a decryption key and wherein the security protocol requires that the second intermediate code is encrypted using a loading key corresponding to the decryption key.

133.   (Currently Amended) A microcontroller operable to execute derivative programs which are derivatives of programs written in an interpretable programming language having a memory and an interpreter, the microcontroller comprising:

(a) the microcontroller operating within a set of resource constraints including the memory being of insufficient size to permit interpretation of programs written in the interpretable programming language; and

(b) the memory containing an interpreter operable to interpret the derivative programs written in the derivative of the interpretable language wherein a derivative of a program written in the interpretable programming language is derived from the compiled version of a program written in the interpretable programming language by applying a conversion of the compiled version including applying at least one rule selected from a set of rules including:

(1) mapping strings to identifiers;

(2) performing security checks prior to or during interpretation;

7

Application Serial No. 10/037,390
Group Art Unit 2124

(3) performing structural checks prior to or during interpretation; and

(4) performing semantic checks prior to or during interpretation.

134.   (Previously Added) The microcontroller of Claim 133 wherein the derivative programs are class files or derivatives of class files.

135.   (Previously Added) The microcontroller of Claim 133 further comprising: the memory containing less than 1 megabyte of storage.

136.   (Previously Added) The microcontroller of Claim 133 wherein the security checks the microcontroller is further comprising:

(c) logic to receive a request from a requester to access one of a plurality of derivative programs;

(d) after receipt of the request, determine whether the one of a plurality of derivative programs complies with a predetermined set of rules; and

(e) based on the determination, selectively grant access to the requester to the one of the plurality of applications.

137.   (Previously Added) The microcontroller of Claim 136, wherein the predetermined rules are enforced by the interpreter while the derivative program is being interpreted by determining whether the derivative program has access rights to a particular part of memory the derivative program is attempting to access.

138.   (Previously Added) The microcontroller of Claim 133 further wherein the microcontroller is configured to perform at least one security check selected from the set having the members:

(a) enforcing predetermined security rules while the derivative program is being interpreted, thereby preventing the derivative program from accessing unauthorized portions of memory or other unauthorized microcontroller resources,

8

Application Serial No. 10/037,390
Group Art Unit 2124

(b) the interpreter being configured to check each bytecode at least once prior to execution to determine that the bytecode can be executed in accordance with pre-execution and post-execution checks, and

(c) the derivative program is checked prior to being loaded into the microcontroller to verify the integrity of the derivative program and loading is performed according to a security protocol.

139. (Previously Added) The microcontroller of Claim 138 wherein the security protocol requires that a particular identity must be validated to permit loading a derivative program onto a card.

140. (Previously Added) The microcontroller of Claim 138 further comprising a decryption key wherein the security protocol requires that a derivative program to be loaded is encrypted using a loading key corresponding to the decryption key.

141. (Previously Added) The microcontroller of Claim 133 wherein the microcontroller is configured to provide cryptographic services selected from the set including encryption, decryption, signing, signature verification, mutual authentication, transport keys, and session keys.

142. (Previously Added) The microcontroller of Claim 133 further comprising a file system and wherein the microcontroller is configured to provide secure access to the file system through a means selected from the set including:

(a) the microcontroller having access control lists for authorizing reading from a file, writing to a file, or deletion of a file,

(b) the microcontroller enforcing key validation to establish the authorized access to a file, and

(c) the microcontroller verifying card holder identity to establish the authorized access to a file.

9

Application Serial No. 10/037,390
Group Art Unit 2124

143. (Currently Amended) An integrated circuit card for use with a terminal, comprising:

a communicator configured to communicate with the terminal;

a memory storing:

an application derived from a program written in a high level programming language format wherein the application is derived from a program written in a high level programming language format by first compiling the program into a compiled form and then converting the compiled form into a converted form, the converting step including modifying byte code operands from references using identifying strings to references using unique identifiers; and

an interpreter operable to interpret such a derivative application in the converted form and derived from a program written in a high level programming language format; and

a processor coupled to the memory, the processor configured to use the interpreter to interpret the application for execution and to use the communicator to communicate with the terminal.

144. (Previously Added) The integrated circuit card of Claim 143 wherein the converting step further comprises:

recording all jumps and their destinations in the original byte codes;
converting specific byte codes into equivalent generic byte codes or vice-versa; and
renumbering byte codes in a compiled format to equivalent byte codes in a format suitable for interpretation.

145. (Previously Added) A method for use with an integrated circuit card and a terminal, comprising:

storing an interpreter operable to interpret programs derived from programs written in a high level programming language and an application derived from a program written in a high level programming language format in a memory of the integrated

10

Application Serial No. 10/037,390
Group Art Unit 2124

circuit card wherein the application is derived from a program written in a high level programming language format by first compiling the program into a compiled form and then converting the compiled form into a converted form, the converting step including modifying byte code operands from references using identifying strings to references using unique identifiers; and

using a processor of the integrated circuit card to use the interpreter to interpret the application for execution; and

using a communicator of the card when communicating between the processor and the terminal.

146.   (Previously Added) The method of Claim 145 wherein the converting step further comprises:

recording all jumps and their destinations in the original byte codes;

converting specific byte codes into equivalent generic byte codes or vice-versa; and

renumbering byte codes in a compiled format to equivalent byte codes in a format suitable for interpretation.

147.   (Previously Added) An integrated circuit card for use with a terminal, comprising:

a communicator configured to communicate with the terminal;

a memory storing:

applications, each application derived from applications having a high level programming language format, and

an interpreter operable to interpret applications derived from applications having a high level programming language format wherein the application is derived from a program written in a high level programming language format by first compiling the program into a compiled form and then converting the compiled form into a converted form, the converting step including modifying byte code

11

Application Serial No. 10/037,390
Group Art Unit 2124

operands from references using identifying strings to references using unique identifiers; and

a processor coupled to the memory, the processor configured to:

a.) use the interpreter to interpret the applications for execution,

b.) use the interpreter to create a firewall to isolate the applications from each other, and

c.) use the communicator to communicate with the terminal.

148.   (Previously Added) The integrated circuit card of Claim 147 wherein the interpreter is further operable to interpret applications derived using a converting step including:

recording all jumps and their destinations in the original byte codes;

converting specific byte codes into equivalent generic byte codes or vice-versa; and

renumbering byte codes in a compiled format to equivalent byte codes in a format suitable for interpretation.

149.   (Currently Amended) A microcontroller operable to execute derivative programs which are derivatives of programs written in an interpretable programming language having a memory and an interpreter, the microcontroller comprising:

the microcontroller operating within a set of resource constraints including the memory being of insufficient size to permit interpretation of programs written in the interpretable programming language; and

the memory containing an interpreter operable to interpret the derivative programs written in the derivative of the interpretable language wherein a derivative of a program written in the interpretable programming language is derived from a compiled for of the program written in the interpretable programming language by performing a conversion including mapping strings to identifiers.

12